APPENDIX H

Technical Plan
Technology Demonstration System
September 10, 1997
(amended)

Contract No. 97-F131000-000

Delivered to the Office of Research and Development

Language Analysis Systems, Inc.
2214 Rock Hill Road—Herndon, VA—20170

1.0 Introduction

This Technical Plan describes LAS's proposed design for the Technology Demonstration System (TDS) and includes a conceptual design, the target hardware platform, operating system, support software and development environment, the name data base and a work plan that provides a schedule for development and implementation.

2.0 Background

The TDS project is the result of the findings and recommendations of the Name Search Research Project, conducted from September, 1995 through June, 1997. The goal of the Project was to determine the utility and feasibility of using phonological information about pronunciation of person names in order to improve the quality of non-exact automatic name searching. Phase 1 of the Project concluded that there was substantial evidence to support the use of phonological information in automatic name searching. Specifically, the conclusions recommended:

- using the International Phonetic Alphabet (IPA) to represent multiple pronunciations of names unambiguously, and
- measuring articulatory similarity of names through phonetic features and processes.

Phase 2 of the Project built upon the results of Phase 1, specifically by:

- expanding, refining and testing sets of IPA rules from Phase 1 to represent multiple pronunciations of Anglo, Arabic, Hispanic and Mandarin Chinese names; test results returned at a retrieval rate of 92%;
- exploring a set of factors that contribute to articulatory similarity, including factors at the syllable level.

Phase 2 recommended the development of a Name Search Technology Demonstration System (TDS) to extend and transfer the phonology-based technology from the Name Search Research Project to a functional, automatic, integrated TDS.

3.0 Environment

The hardware and software environment are well defined. It is a simple environment that is geared to flexibility and performance. In other words, LAS does not intend to introduce complications by using resource intensive and/or expensive support software or hardware. TDS will be built using the following hardware and software components:

- Standard, high performance Intel-based laptop computers. By purchase time, we expect to configure the machines as follows:
    - Intel Pentium, Pentium Pro or Pentium II CPU;
    - 160 to 512 Mb of memory (160 is the current maximum);
    - 3 Gb of disk storage;
    - High speed CD-ROM (8x minimum);
    - Standard Ethernet network card for high speed data transfer;
    - High resolution monitor with a bright clear screen.
- Windows 32-bit operating environment and development software:
    - Windows 95 or Windows NT 4.x (depending on the processor available);
    - Microsoft Visual C++ version 5.x (for development only);
    - Microsoft Access (support table maintenance);
    - Custom data storage techniques maximizing memory usage (no RDBMS);
    - Multi-threaded architecture to begin displaying responses within12 seconds.

TDS will run on a top-of-the-line standard laptop computer, a suite of custom developed executables and dynamic link libraries (DLLs) and standard end user software (i.e., MS Access, Excel, etc.). There will be no special or extraordinary system or software requirements.

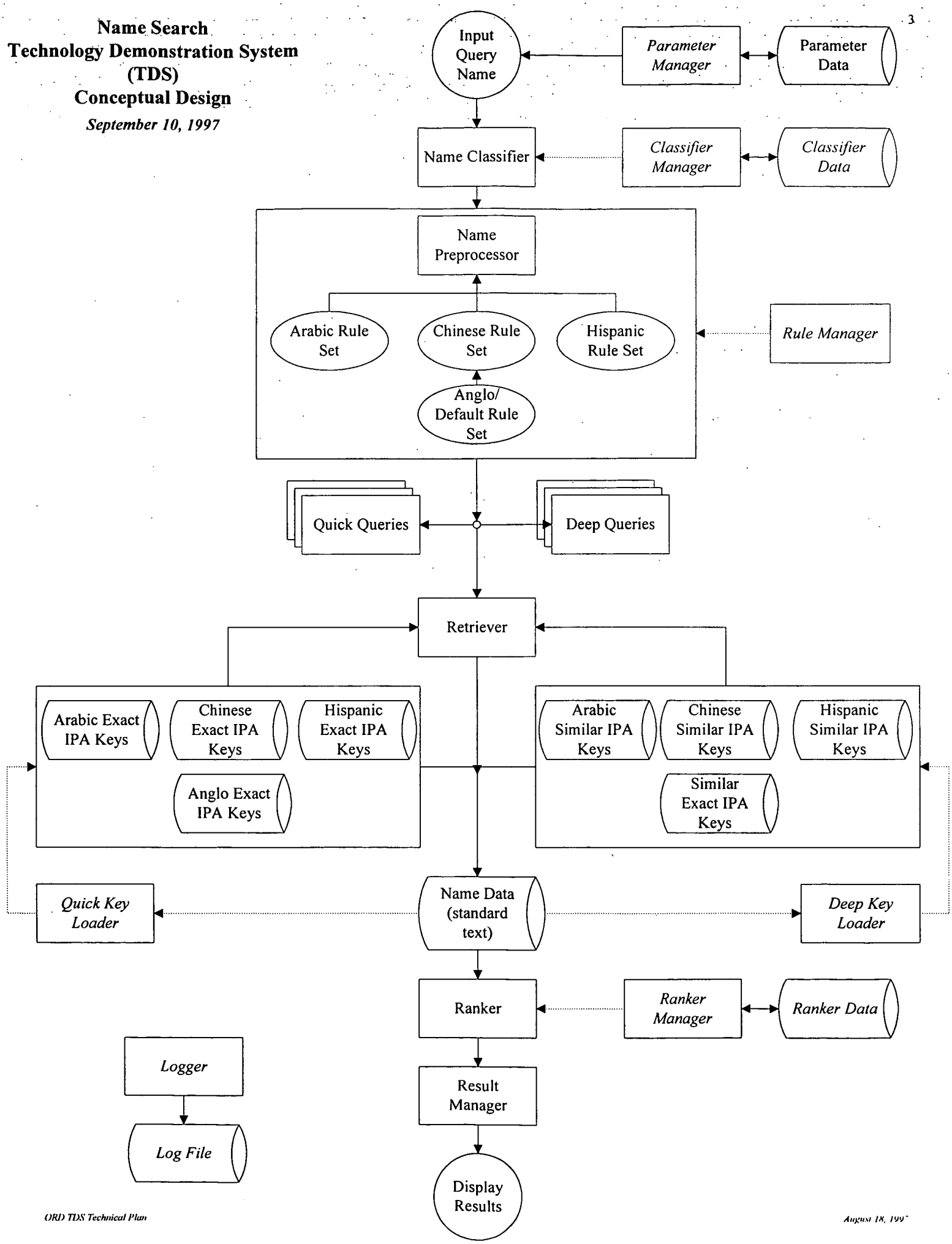
4.0 Conceptual Design

The following diagram gives an overview of the system LAS intends to build. It is based on previous documents developed by the sponsor and LAS and represents research done to date. It identifies the major components and processes to be included in TDS; however, it is a preliminary design, and is subject to change depending on the outcome of further research and time restraints. After the diagram, each of the components and interactions between them is described in further detail.

In the chart, components that are identified in italics are support functions not intended for standard use. They are, however, necessary for LAS to build, tune and test the system.

Name Search Technology Demonstration System (TDS) Conceptual Design

September 10, 1997



Input Query Name - A simple Graphical User Interface (GUI) will be built to allow the user to enter a name that will be compared to the name data base.

Parameter Manager - LAS will provide options so the user can tune or limit the search. These options will be controlled through the GUI and stored in a **Parameter Data** set. Parameters currently being considered include:

- exact only (fast) or exact and similar matches;
- level of similarity (loose or tight);
- culture specific matches (Anglo, Arabic, Chinese and/or Hispanic);
- number of returns (maximum/default = 145);
- bypass name classification.

Note that it will not be necessary to set parameters to perform a name search. Default parameters will be used in the event that the user goes directly to the name check screeen.

Name Classifier - The name classifier determines whether the ethnicity of a name is Arabic, Chinese or Hispanic. The LAS name classifier uses a data base of contiguous letter pairs (digraphs) and triplets (trigraphs) that has been statistically analyzed to rank digraphs and trigraphs according to ethnic origin. With this information, it calculates a score for each culture that shows the probability of the name being Arabic, Hispanic and/or Chinese The highest positive score will determine which non-Anglo algorithm to use in addition to the standard Anglo algorithm. Note that it is possible for all scores to be negative, in which case only the Anglo algorithm will be used . This component will be based on an existing system developed in Clipper by LAS that will be converted to C++ to better interface with the other components.

Classifier Manager - This is a simple interface necessary to apply values to the digraphs and trigraphs according to ethnicity. Most likely, LAS will use a standard data base package to manipulate classifier data (i.e., MS Access). Note that the existing classifier data base is already returning adequate results. Improvements will be made if time and resources permit.

Name Preprocessor - At a minimum, this component will convert the input name into one or more IPA representations. Almost certainly, it will generate numerous variants based on different phonetic representations that will be passed to the **Retriever**. Furthermore, additional information about the query name will be necessary in order to use the similar search keys (i.e., name length, syllabic structure, etc.).

Rule Sets - Four rule sets will be used to convert Roman character representations of names into IPA representations. The default rule set, Anglo, will always be used; the other three, Arabic, Chinese and Hispanic will be used if the name is classified as belonging to one of these ethnic groups and the user has specified that other ethnic variations are to be used. These rule sets will be based on the work done in previous projects. The Anglo rule set will need considerable

modification to support Anglo pronunciations of non-Anglo names. They will be maintained by a **Rule Manager**, that allows LAS to build and modify rule sets as necessary.

Quick Queries - To ensure that the 12 second initial response time requirement is met, LAS intends to segment and multi-thread searches of the name data base. Quick queries retrieve those records that contain the same IPA characters or the same IPA consonants with a vowel place-holder, or the same IPA consonants. The ultimate retrieval scheme will be determined by further research. This approach will allow TDS to pass a small subset of data to the Ranker and begin returning most of the "best" names quickly. Note that this scheme does not consider differences in name length (i.e., insertion and deletion). The output of the quick query component will be a list of IPA representations that the **Retriever** will use to extract records for evaluation by the **Ranker**.

Deep Queries - By far the most difficult problem to solve, deep queries will allow TDS to subset the name data base into phonetically similar sections and account for varying levels of name and possibly syllable length. They must consider the insertion and/or deletion of IPA characters and the proximity of different IPA characters based on the number and importance of features they have in common (e.g., "p" and "b" differ by only one phonetic feature). Almost certainly, deep queries will include all names retrieved by quick queries. If performance is acceptable for deep queries, the quick query logic may become unnecessary. The output of the deep query component will be a list of IPA representations that the **Retriever** will use to extract records for evaluation by the **Ranker**.

Retriever - This component accepts query lists from the query preprocessors and passes subsets of the name data base to the **Ranker**. Operating simultaneously with the query components, it processes query lists in the order that they are received. Each input list will be identified as a "quick" or a "deep" list so that the component can choose the proper key set to use to generate the output list. Once the subset is determined, the retriever will build a list or a range of records to be passed to the **Ranker**. This list will contain the IPA representation used to retrieve the record, the actual Roman character representation of the name and the rule set used to return the name.

Quick Keys - Each name in the data base will generate one or more IPA representations of the Roman character version. Each rule set can generate different IPA representations. All representations will be stored in the **Quick Key** data set that will point to the name that generated the particular version. Furthermore, quick keys will be tagged as belonging to the rule set that generated the representation.

Deep Keys - This data set will consist of keys that contain IPA representations, IPA similarity, name length and possibly, syllabic information. It will be designed to allow for subsetting of the name data base into names that are potentially similar to the query name. It must overcome the two major problems in determining name similarity: sounds can be mispronounced (Pine = Bine) and names can be substrings of each other (McDonald = Donald). A key area of research that must be resolved early in the project is the use of indices to represent similar IPA characters (one character to represent "b" and "p").

Name Data - This component represents the raw data provided by LAS internally for development and ultimately by the sponsor for the production version of TDS. Each name will be stored in its Roman character representation and will be identified by a record ID. These ID's will be used to tie the raw name to the quick and deep keys.

Key Loaders - Batch programs will be developed that take an ASCII text file of names as input to generate the **Name** , **Quick Key** and **Deep Key** data bases. This program will use the **Name Classifier** and **Name Preprocessor** to generate keys and build or rebuild these data bases. It will edit the names and produce a summary statistical report and a detailed error report showing any abnormalities encountered (i.e., invalid length, invalid characters, etc.).

Ranker - This component processes a list of candidate records generated by the **Retriever**. The list will consist of records containing the IPA key that returned the record, the rule set used to generate the IPA key and the actual name. The Ranker will sort the names in order of match quality based on parameters set in the **Ranker Manager** data set. Output will be passed dynamically to the Result Manager for real-time display to the user. Ranking methods will be based on schemes developed in phase 2 of the phonology project (regular expression intersection and the "voter scheme"). It will also consider the rule set used to regularize the input name into IPA representations.

Ranker Manager - This is an optional component that will allow LAS and/or the sponsor to rank returns according to different sort schemes. As mentioned above, the ranking schemes will be based on previous work: regular expression intersection or a voter scheme, and possibly, non-phonetic schemes such as: Soundex, digraph analysis, edit distance methods, etc.

Result Manager - This component will accept input from the **Ranker**, and maintain a deduped, sorted list based on the parameters set by the **Ranker Manager**. This list will be passed to the GUI for display to the user, and it will be managed dynamically so that the list is constantly being updated as results are processed by the **Ranker**.

Display Results - This component is the output side of the GUI. It displays the list produced by the **Ranker** for viewing and other manipulation (printing, saving to a file, etc.) by the user. The outputs are maintained by query name and are updated dynamically as results are returned from the **Ranker**. In addition to the ranked list of names, the GUI will also display information on why a particular name was chosen and will be given a score that relates to names above and below it on the list.

Logger - This component will be a development and debugger tool for LAS to determine how well TDS is working, and to aid in testing and problem resolution.

5.0 Name Data Base

The ultimate target for TDS is a sponsor data base consisting of 3 million unique name segments (i.e., "John" and "Fitzgerald", not "John Fitzgerald"). LAS must generate a similar data base of name segments since the sponsor data base is classified. To do this, LAS will take advantage of numerous resources that will be used without compromising the privacy and sensitivity of the data. That is, only name segments will be extracted from these sources. It will be impossible to tie the TDS names to the source data base. Sources to be used include:

- Visa Lookout Data from the Department of State;
- Passport Lookout Data from the Department of State;
- Census Data from the Department of Commerce;
- Phone Book data from commercial sources;
- Known variant lists.

Should the above sources fail to generate 3 million unique name segments, LAS will resort to generating variations by programmatically manipulating letter variations (i.e., "ck for "ch", "e" for "i", etc.). Currently, LAS has processed 20 million names,which have generated 1 million unique name segments.

Crucial to the successful completion of TDS is an opportunity by LAS to evaluate sponsor data as soon as possible. While not in the Statement of Work or the Project Plan, LAS feels it is advantageous to the sponsor to allow LAS to gain access to the sponsor name data base as soon as possible. While no problems are expected, it is prudent to verify this assumption as the success of TDS is ultimately dependent on the ability to successfully integrate sponsor data.

6.0 Work Plan

Please note that this section of the Technical Plan has been copied in entirety from the Project Plan previously submitted. Attachment A to this plan is a Gantt chart with a Work Breakdown Structure that describes the schedule of development LAS intends to follow. The rest of this section describes the major events in the Gantt chart.

The schedule for the development of TDS spans eight months and consists of four major phases:

- **Planning** - One month to generate project and technical plans.

- **Phase 1 Development** - Three months to resolve research issues, determine a strategy to find "similar-to" names, define and validate linguistic search techniques, and produce a limited version of TDS for an early look test.

- **Phase 2 Development** - Three months to expand phase 1 into a fully functional system to include expanded rule sets that enable TDS to accommodate Anglo pronunciations of foreign names and native pronunciations for Arabic, Hispanic and Chinese names.

- **Implementation** - Four months (with three months overlapping the development effort) to procure two laptop computers, install the system for the sponsor, provide training, and document the results of the project.

- **Maintenance** - Four months to modify, upgrade and correct TDS at the direction of the sponsor.

Each phase concludes with a specific set of deliverables (both internal to LAS and formal deliveries to the sponsor). There is some flexibility in the schedule, however, the dates set for sponsor deliveries are firm.

6.1 Phase 1 Development

The purpose of phase 1 is prove that LAS can develop a viable name search system based on phonetics. The goals are to produce a complete design for TDS and develop a prototype system. Although limited in functionality, the prototype must be complete enough to pass an early look test based on a test plan generated by LAS. The sponsor has the ultimate authority to decide whether or not the prototype justifies further development. Phase 1 consists of the following tasks:

- **Research** - Previous work by LAS has generated many working theories and prototypes/work benches. All of this work must be analyzed further to determine which theories are best applied to TDS. In early September, when this task is scheduled to conclude, LAS will know how all of the major components of TDS will work and will have a conceptual design document that drives further development. In addition, LAS will deliver input specifications for data to be loaded into TDS.

- **Development** - Based on the outcome of the research task, LAS will develop the first limited version of TDS. Ideally prototypes developed during the research task will form the basis for this version of the system. In addition, the Linguistic team will continue to refine their research from the previous period and provide guidance to the Technical team.

- **Test Plan** - During research and development, a test plan will be developed. First, requirements will be culled from existing documentation and results from the research task. Then, these requirements will be used to develop test scripts. There are four major areas to be tested: functionality, performance, retrieval accuracy and ranking accuracy. The test plan is a deliverable required by the contract.

- **Build and Test** - A week has been reserved in the schedule to integrate the output of the development task, after which there are three weeks to execute the test, make any corrections and document the results.

While subject to change, the plan calls for the first version of TDS to contain the following features:

- A fully functional name classifier that can identify Arabic, Chinese and Hispanic names. The name classifier will be ported to C++ from LAS's already developed PC-NAS system that is currently written in Clipper.

- Name processors for Anglo and Arabic names. Note that in this phase, the Anglo name processor will not include the extended rule set for atypical Anglo names.

- A fully functional name data base with a key structure that accommodates both IPA exact match and phonetically "similar to" searching. A program to load raw data into the name data base will also be produced during this phase. The name data used will be obtained from LAS resources.

- A search engine that when given a query name and it's ethnicity will search the name data base and provide a list of matches.

- A limited version of the ranker with a sorting algorithm to be determined during development.

- A limited graphical user interface (GUI) to allow for evaluation of the TDS.

6.2 Phase 2 Development

Phase 2 provides three months to complete the development of TDS. Currently, the features to be developed in this phase are:

- Develop the Hispanic and Chinese name processors.

- Extend the Anglo name processor to include rules for atypical Anglo names and pronunciations.

- Complete the Ranker to include a sort algorithm with additional sorts as deemed useful.

- Finalize the GUI to include all features required by the sponsor and/or deemed desirable by LAS.

- Finalize the TDS documentation to include a simple user manual and descriptions of all algorithms.

Phase 2 culminates in the execution of an acceptance test with time built in for bug fixes and test documentation.

6.3 Implementation

The final task is to deliver the system to the sponsor. LAS will purchase, test and configure two high-end laptop computers, load sponsor data into TDS, provide training, and write a final report.